The genesis of a knowledge-based expert system

by J. A. Voelker
G. B. Ratica

This paper discusses the genesis, development, and testing of a knowledge-based expert system called the Contract Support Services Consultant. This system aids in the process of estimating, bidding, and preparing agreements for certain services required for the rearrangement, relocation, discontinuance, and reinstallation of IBM equipment. The goals in developing the prototype of the Contract Support Services Consultant were to capture the knowledge and experience of experts and make them available to nonexperts in a convenient, consistent way. The result is a system that improves the accuracy, consistency, and timeliness of the estimates and bids.

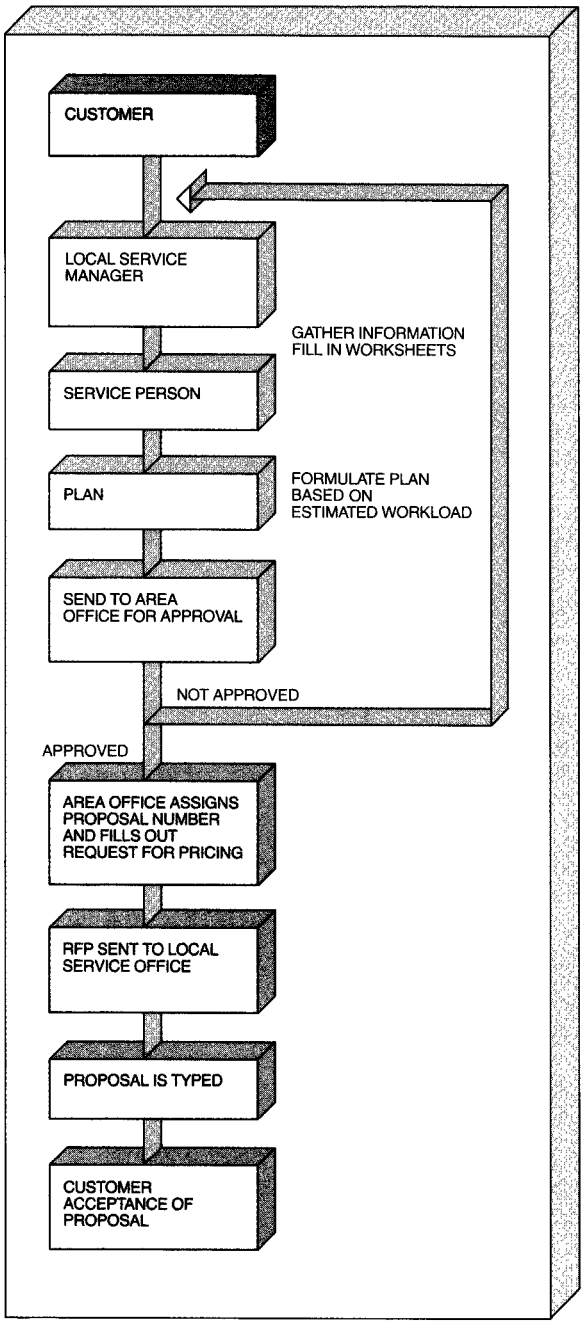
Contract Support Services are fixed-price services offered by the National Service Division for specified equipment and/or programs, in addition to those services provided under IBM's Lease, Rental, Maintenance, Purchase, or License Agreements. In this paper, we define a subset of these services, describe the existing process of estimating their cost, point out some problems with this process, and describe our approach to solving these problems.

We begin by giving some background relevant to Contract Support Services. Customers expect consistent estimates and bids on services performed for the same job. Thus two different estimators should not come in with two widely differing bids, unless the job factors have changed. Customers further expect these bids and/or estimates to be accurate, and they expect IBM to appreciate the value of their time. Thus a timely bid or estimate is important to a customer when planning changes that disrupt the normal work schedule.

Typical Contract Support Services involve the rearrangement, relocation, discontinuance, and reinstallation of IBM equipment. *Rearrangement* services normally required when IBM equipment is rearranged within the same room are disassembly, to the extent necessary for rearrangement within the same room, reassembly and the connecting of cables, and testing the equipment. *Relocation* services are normally required when IBM equipment is moved or relocated within the same building or when IBM equipment is relocated outside the building, and discontinuance and installation services are not required. These services include disassembly, to the extent necessary, reassembly and the connecting of cables, and testing the equipment. *Discontinuance* services are normally required when IBM equipment is discontinued to prepare it for shipment to another location. Included are disassembly, identification of items that are considered part of the machine and are to be shipped with it, placing manuals, diagnostics, covers, cables, small units, and miscellaneous hardware in shipping containers and preparing for shipment, and providing technical information and guidance for packing. *Installation* services are normally required when IBM equipment is reinstalled after being moved from another location. These services include provision of technical information and guidance for unpacking, assembling and con-

© **Copyright** 1986 by International Business Machines Corporation. Copying in printed form for private use is permitted without payment of royalty provided that (1) each reproduction is done without alteration and (2) the *Journal* reference and IBM copyright notice are included on the first page. The title and abstract, but no other portions, of this paper may be copied or distributed royalty free without further permission by computer-based and other information-service systems. Permission to *republish* any other portion of this paper must be obtained from the Editor.

Figure 1 The Contract Support Services estimating process



necting of cables, testing and reconfiguration of diagnostic tests, and removal of IBM internal packing material.

The majority of services available under Contract Support Services are also available from IBM on an Hourly Service basis. However, when service is provided on an hourly basis, the customer is subject to recurring or substantial billing which is difficult to include in budget planning. Contract Support Services makes these services available on a predetermined fixed-price basis.

If the customer requests an estimate, the current practice is for the local service branch office to assign a field service person to estimate the work required to perform the requested services. The estimated workload is used to formulate a plan for the work. This plan must be reviewed for accuracy and consistency with historical data and approved by the next higher level in the reporting structure at an Area office. If the plan is not approved, the local service manager must submit a new plan. If it is approved, the area office assigns a number and generates a Request For Pricing (RFP) that is returned to the originating office. The originating office then prepares an agreement for submission to the customer. This procedure generally requires an average of ten days to produce an agreement. A diagram of this process is shown in Figure 1.

The estimation of the work and the preparation of the bid involve rather complex procedures. Machine type and type of service are primary considerations in determining the cost. With over three hundred unique machine types installed in the field, the possibilities increase enormously when the machine type is combined with the type of service requested and the multitude of other factors that must also be included. For example, units may have to be unbolted and separated for a relocation within the same building but not for rearrangement within the same room. Such factors obviously affect the amount of time required to perform the service.

Other less obvious factors must also be considered in the estimate process. For example, type of floor, clearance for interconnection cables under a raised floor, number of cables, extent of diagnostic test reconfiguration, and length of cables are some of the variables that must be considered when preparing the estimate. Because each installation location is different, it is difficult to find other installations to serve as models. Ordinarily, finding common denominators in such instances would help resolve these problems, but in most cases they do not exist. Likewise, there is no common repository of accu-

mulated knowledge or expertise for the field personnel to draw on. They must rely on their individual memories, common sense, and experience for each situation. Yet each new situation is unique.

Although there are experts in the field, they tend to become more and more specialized in their knowledge of machine types. Also, experts are not always available to their peers or to those estimators whose experience and knowledge are less specialized.

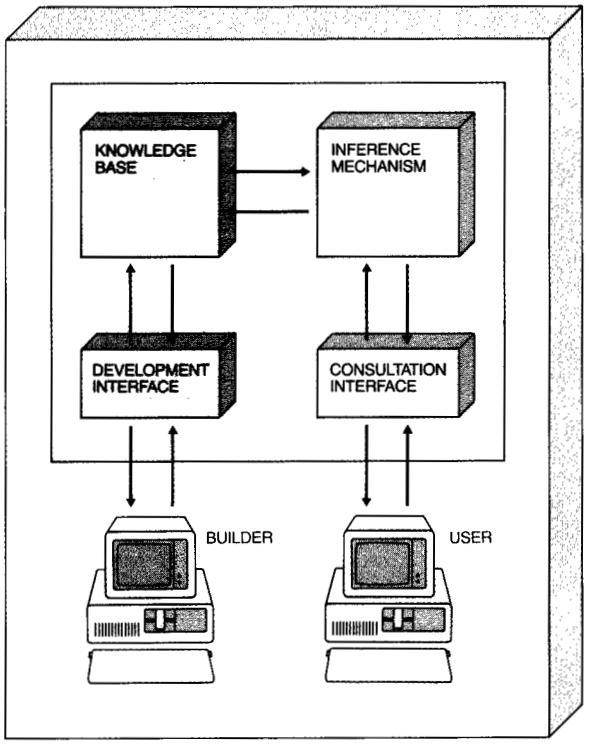
Our goal has been to find ways to make estimates more consistent, more accurate, and more quickly prepared. As part of its mission, our department (ISG Service Research, Knowledge Based Systems) has been studying knowledge-based systems and building prototypes for several years as a possible way of solving problems. On the basis of our experience, it has seemed to us that a knowledge-based system could be developed to make it possible for nonexperts to deal with complex problems involving a large body of knowledge gleaned from experts.

Prior work

Before presenting our research, we briefly discuss the prior work of others, as well as that of our department, that led us to choose a knowledge-based expert system as an approach to improved Contract Support Services. A block diagram of a typical knowledge-based expert system is shown in Figure 2. The figure shows a body of expert knowledge, a means of interpreting that knowledge (called an *inference mechanism*), and a means of making the expert knowledge available to a nonexpert (called a *consultation*).

The MYCIN study. One of the earliest systems we studied was MYCIN, which was developed at Stanford University, starting in 1972, by researchers who were interested in using computer programs to interpret medical data.[1] This system was designed to provide expert consultation and advice to physicians, who were not specialists, on the diagnosis and treatment of infectious diseases. It includes a large body of knowledge in the form of IF-THEN rules gathered from experts, as well as an inference mechanism to interpret the knowledge. The MYCIN system can also explain the line of reasoning it uses to reach a particular conclusion, as well as the reason for asking a particular question. This is the approach that a human expert typically takes in a consultation with a nonexpert. Because no body of knowledge is static,

Figure 2 A typical knowledge-based expert system



MYCIN has the ability to allow the updating of the knowledge base. Experience with MYCIN and other knowledge-based systems has shown that the abilities to answer questions, ask questions, explain the reasoning in both instances, and be updated are essential to the development of a useful, usable consultation system.

Essential-MYCIN (EMYCIN). In MYCIN, which was built from scratch, the program code for the knowledge base and the inference mechanism is tightly intertwined. It became apparent to the developers that some of the programming would be the same for other expert systems and need not be rewritten each time. This suggested the possibility of developing a tool that did not require expert systems based on other knowledge bases to be built from scratch.

This led to Essential-MYCIN (EMYCIN), which is an *expert-systems shell* or tool for building expert systems. EMYCIN is based on the inference and reasoning core of the MYCIN program, but it is independent of MYCIN's domain-specific knowledge base.[1] This shell

contains the inference mechanism, the knowledge-base construction aids, and the program that conducts the consultation.

We were intrigued by the potential for the use of expert systems in the service community, and, together with members of the IBM Palo Alto Scientific Center, we entered into a joint study with Stanford University during 1980–1982. The purpose of the study was to determine the feasibility of applying expert-systems technology to computer failure diagnosis. As a test, we undertook the development of a prototype diagnostic system.

We were intrigued by the potential for the use of expert systems in the service community.

The Diagnostic Assistance and Reference Tool project. The domain chosen was Direct Access Storage Devices (DASD) and a DASD controller, the IBM 3880. The consultation system was built using EMYCIN and was installed and used for validation in the IBM San Jose DASD Field Support Center.

The Field Support Center provides second-level support to field service people via telephone. The support includes diagnostic assistance and referral to product engineering when appropriate. The diagnostic system was installed to provide problem determination assistance for difficult-to-diagnose symptoms. The system was designed to isolate failures to one of five areas: the central processor hardware, the central processor software, the 3880 controller hardware, the 3880 controller software, or the attached disk drives.

Because of the software characteristics of EMYCIN, the diagnostic system could not be used on the workstations normally used by the support center personnel. Therefore, additional workstations were installed adjacent to the regular workstations. These additional workstations were of a line-at-a-time type of display, rather than the full-screen displays normally used, and the processing for the diagnostic system was not done on the regular support center system. The processor used was at a university, remote from the support center, and was linked to the workstations by telephone lines. These conditions resulted in inconsistent response times and the need for the support center people to deal with two different displays and two different protocols. These factors turned out to be very important, and we will discuss them further later in this paper.

Our experience in building the DASD diagnostic system using EMYCIN indicated that the inference mechanism could indeed be used for knowledge about computer failures as well as infectious diseases. In addition, the use of English-like rule statements to interface to EMYCIN allowed people who were not experienced programmers to create the knowledge base. This English-like interface also simplifies the maintenance and enhancement of the knowledge base. Also, our use of the diagnostic consultant convinced us that expert-systems technology showed promise for the service community. We found that the consistency of the advice provided to the field had improved, and the system also proved to be useful as a tool on which to train new people in the support center because of its explanation facility. When it was used, the prototype also demonstrated that it reduced dependence on product engineering, reduced the number of customer interruptions, and reduced the total number of hours required to solve problems.

However, it became apparent that factors other than the system's effectiveness were more important than we had realized. These were related to the way that the system fit into the workplace. We found that there was reluctance to use the system during an assistance call from the field. The reasons seemed to be that the support center person had to use another workstation, in addition to the normal workstation. In an environment where the person's job was to provide telephone assistance as expeditiously as possible, use of the system tended to prolong telephone calls. Also, use of the system was viewed as an additional task, not an aid to job performance. These results emphasized the importance of carefully integrating an expert system into the existing workplace and provided us with an important lesson for later expert-systems development work.

Mindful of this lesson, a group at the Palo Alto Scientific Center, with our collaboration, began the development of an expert-systems shell, similar in

some respects to EMYCIN, that would allow expert systems to be developed that could be delivered on the workstations that our people normally used. This shell was used for some of the early work on the next system we developed (discussed later in this paper) and was the forerunner of the IBM Program Offering called Expert System Environment.

The Program Information Network Expert project. At about the same time that the shell just mentioned was being developed, we began work on another prototype application, the Program Information Network Expert. Whereas the domain of the Diagnostic Assistance and Reference Tool was computer hardware, the focus of the Program Information Network Expert was in the software area. Remembering the reluctance of people to use the Diagnostic Assistance and Reference Tool when they were involved in a telephone consultation, we chose a task for the Program Information Network Expert that did not have real-time implications. The Program Information Network Expert is an expert consultant for use by Program Service Representatives (PSRs) at the IBM Support Center in Chicago. This system was designed to provide expert guidance in the process of editing and correcting records of problems that are stored in a large computerized data base. This process is called Program Information Network authoring, and the task is performed when PSRs are not occupied with their principal duties of consulting on software problems over the telephone.

In the first prototype of the Program Information Network Expert, the authoring assistance consultation was delivered via a specialized workstation that the users either visited or moved to their desks. If the PSR were to visit a workstation that was remote from the normal workstation, it was not possible to switch quickly from authoring to the primary responsibility of helping a customer solve a software problem.

When the specialized workstation was brought to the PSR's regular workplace, it did not fit smoothly into the work environment because of the extra space it required and the noise it generated.

Although the Program Information Network Expert proved to be effective in improving the results of the authoring process, we again found that the issue of fitting the tool to the existing work environment was an important one for usage and acceptance. The next version of the Program Information Network Expert was delivered via Expert System Environment/VM on the PSRs' normal workstations. We expect that this better fit of the tool to the existing work environment will lead to increased acceptance and use.

The Contract Support Services Consultant

The study of expert systems is part of the technical training given upon joining this department. As a result, we saw a possible fit between the Contract Support Service estimating process that we had encountered in the field and expert-systems capabilities. This led to the development of the Contract Support Services Consultant. The consultant is an expert system that is used by field service managers in local field service branch offices to prepare estimates and bids for contract support services.

The consultant is an expert system that is used by field service managers to prepare estimates and bids for contract support services.

Each service branch office normally receives information services support from an information center at the next higher level in the reporting structure. The local service managers regularly access these services from personal workstations. We felt that if we could deliver the consultation via the manager's regular workstation, it would go a long way toward overcoming one of the major hurdles we had faced earlier in the acceptance and use of expert systems—the fit of the tool to the work environment. We therefore chose the shell developed at the Palo Alto Scientific Center and later its successor, Expert System Environment/VM, as the shell to develop the Contract Support Services Consultant.

Building the Consultant. Some of the other reasons why we made this choice had to do with the ease of use of the shell. The knowledge base is created by entering IF-THEN rules in a language very much like English, which makes it possible for people who are

not experienced programmers to build the knowledge base. It also allows the consultant system to be maintained and enhanced by people without extensive programming skills.

Expert Systems Environment/VM uses powerful editors to provide semantic, syntactic, and cross-reference checking as the knowledge base is being built. This identifies errors and gives the builder a chance to correct them immediately, rather than waiting until the knowledge base is complete and correction is much more difficult.

In order to acquire the knowledge to build the system, we interviewed people who were experienced in performing the work of rearrangement, relocation, discontinuance, and reinstallation. We then encoded their knowledge as IF-THEN rules.

The knowledge base. The knowledge base is composed of three types of objects: parameters, rules, and focus control blocks. *Parameters*, which are the domain facts, are similar to variables in traditional programming languages. *Rules* describe the relationships among parameters, and *focus control blocks* allow the knowledge base to be broken into components and provide for control of the consultation. The following is a sample rule:

```
RULE: UNDER_FLOOR_SPACE_NO

  if under_floor_space is 'insufficient'
  then
  extra_time = .05
```

This rule, named UNDER_FLOOR_SPACE_NO, assigns a value of .05 to a parameter named extra_time if the parameter named under_floor_space is found to have the string value 'insufficient'. The purpose of the rule is to allow .05 or 5% extra time in the estimate for cable relocation when there is limited room for cables under the raised computer room floor. The fact that under_floor_space was insufficient would have been determined by asking the user a specific question during the consultation. The condition of insufficient room under the raised floor is the something that a nonexpert might well overlook.

The example also shows the use of descriptive names for the parameters and rules, which we believe is important in maintaining the knowledge base. Also illustrated are parameter values (extra_time) as a

Expert System Environment

Suppose you—as an expert in your field—want to capture your professional knowledge and experience in the form of a computer application to assist those who are just coming into your field. You are contemplating an *expert system*. How would you build it? Where would you begin?

There are tools developed by IBM that are to the expert or the *knowledge engineer* what the spread sheet is to persons in business and engineering. Those tools are Expert System Development Environment and Expert System Consultation Environment—an *expert system shell*—a general purpose system for constructing and executing expert system applications.

Using these tools, it is possible for *experts* to insert information easily and create an expert system in the *domain* of your particular profession. For complex problems, the services of a knowledge engineer may be valuable to aid the expert in generating and structuring the *knowledge base*.

Expert System Development Environment and Expert System Consultation Environment grew out of the observation that experts or knowledge engineers often experience difficulty in creating an expert system, and it seemed possible to design an environment to help them. The development environment is designed to provide a base for the construction of a variety of specific expert system applications, and the consultation environment allows the user to execute those expert system applications on different processors.

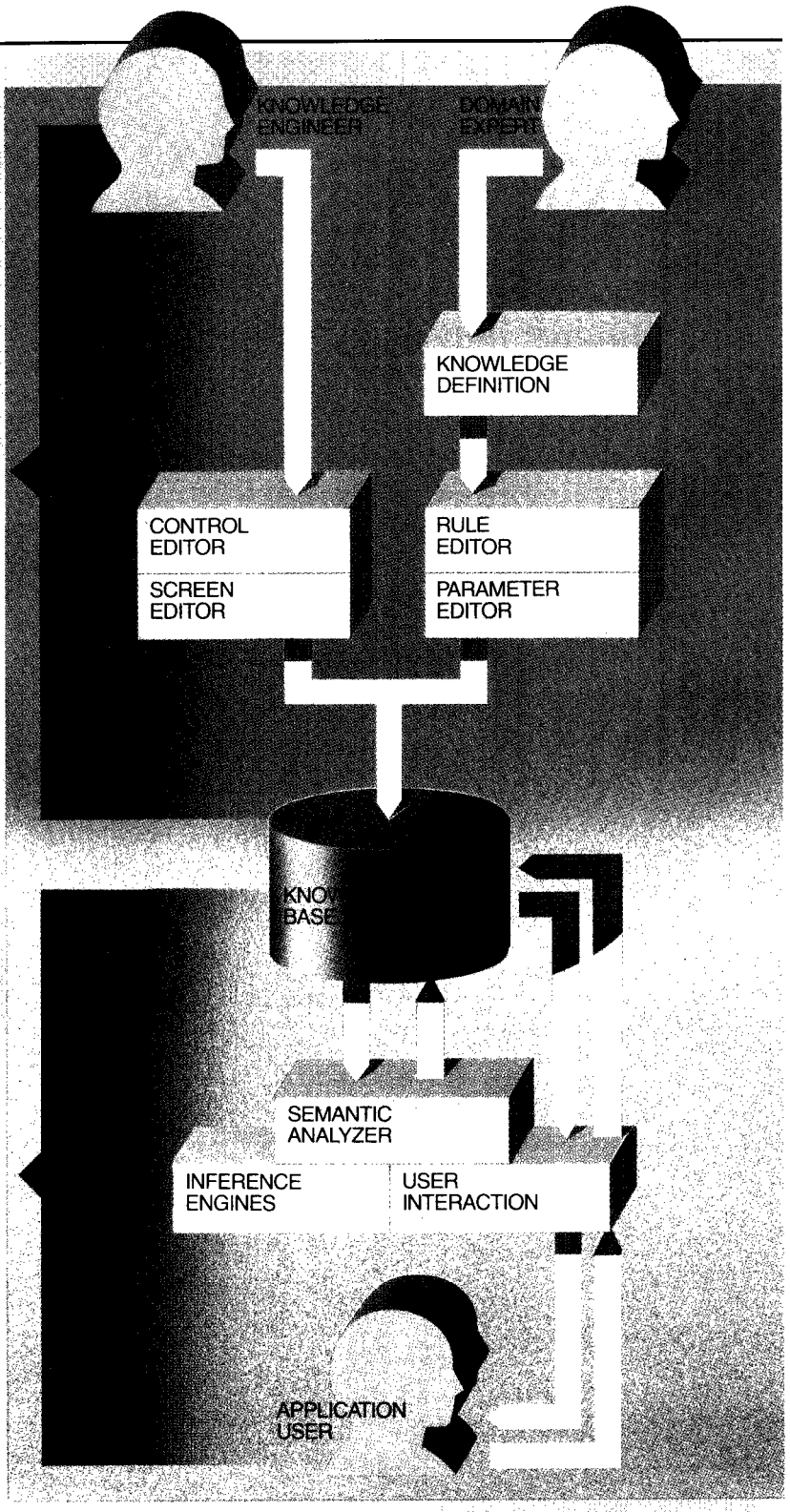
The development environment incorporates a set of procedures that can be used to design a variety of applications. It can be thought of as an empty system or shell into which experts or knowledge engineers insert their own rules to define the knowledge base. They choose a reasoning method to select the inference engine that will apply the rules. The result is a specific expert system application ready for the user to

interact with via Expert System Consultation Environment.

The development environment incorporates *editors* that allow the expert or a knowledge engineer to create the knowledge base in the form of English-like *rules, parameters,* and *controls.* There is also a *screen editor* to allow developers to design their unique screens for multiple questions, explanations, and the display of results. The consultation environment includes a set of basic *inference engine* processing functions. These include both backward chaining (to obtain a value for a desired goal by working backward from the goal to find given data) and forward chaining (to proceed forward from given data). Using either method, the inference engine begins to make inferences, draw conclusions, or carry out actions using the *semantic analyzers.*

Current areas of application include computer software and hardware diagnosis, the configuring of computers, the generation of service agreements, and a variety of financial, banking, and military applications. Both the development environment and the consultation environment are written in Pascal and run on System/370 under the VM/CMS and MVS/TSO operating systems. Interfaces have been provided to allow access to data files and user-written programs. Further information can be found in the Expert System Development Environment and Expert System Consultation Environment General Information Manual (GH20-9597).

T. Jay Ashford



consequence of a rule and as a result of a user response (under_floor_space). Parameter values can also come from external data routines or take a default value.

The consultation. During a consultation, the user is first asked such administrative questions as customer name, address, and so forth. The system then asks questions to determine specifics of the work to be estimated, e.g., type of service, machine types and quantities, and environmental conditions. The consultation is controlled by the inference mechanism or inference engine, as it is sometimes called. The inference method used in the consultant is goal-driven backward chaining. This is specified by control language coded in the consultant that identifies the goal(s) of the consultation and causes the inference engine to reason backward to determine the facts needed to establish the goal(s). Many of these facts are provided by the user during the consultation; others are stored in the knowledge base, and still others are the consequences of rules.

The main goals. The main goals of the consultant are to estimate the total charge and the time required for the work requested. Control language is used to identify them to the consultant as goals and to direct the inference engine to determine their value by backward chaining. This process is illustrated in the following example, in which quantities are identified by the parameters "charge" and "total_real" and are established by the rule COMPUTE_CHARGE:

```
RULE: COMPUTE_CHARGE

if  total_cabling_time  is known
    and packing         is known
    and premove         is known
    and coolant         is known
    and old_frame       is known
    and standby         is known

then

charge = rate * realtime * num_of_machs
         + (total_cabling_time * rate)
         + (total_diag * rate)

and

total_real = realtime * num_of_machs
```

Here, the text of the rule has been formatted to make reading and understanding easier; this type of formatting is not required by the expert-systems shell but is helpful for maintenance and enhancement.

The parameters shown in the if-part of the rule (the *premise*) all have an effect on the time required ("realtime") to perform the requested service on a given machine. For example, "packing" represents the installation of optional internal packing materials before a machine is moved. If the service being estimated is rearrangement within the same room or installation, the packing option does not apply, and the consultant contains rules that set the value of "packing" to no, without querying the user, and no time is added to "realtime" for packing. If the service is to discontinue or relocate, the consultant asks the user a question to determine whether the customer has chosen the packing option and sets the value of the parameter accordingly.

A user survey administered early in the test showed a high degree of acceptance of the system.

Similarly, the values of the parameters in the then-part of the rule (the *conclusion*) are needed to calculate the "charge" and the "total_real," and the backchainer executes rules until the necessary values have been established to perform the calculations.

Because the consultant has then established the goals identified in its control language, the consultation is ended.

Other features. Besides calculating the time and charge for a given estimate, the consultant prepares a worksheet for the manager to use in schedule and resource planning, and it can create and electronically send a copy of the worksheet and estimate to the next level in the reporting structure for review. The consultant can also print out an agreement for submission to the customer. Thus, the consultant assists the estimator, but it is not intended as a replacement for a human estimator. The recommendations of the knowledge base may be overridden.

Present status. The consultant is currently being tested and evaluated in three geographic areas for

three factors: user acceptance, accuracy and consistency of the estimates, and impact on the host system. A user survey administered early in the test showed a high degree of acceptance of the system, with few complaints. When enough data have been accumulated, the estimates produced by the consultant will be compared with actual totals for the services performed, and the system will be fine-tuned, if necessary. At that time, we plan to stop reviewing the estimates.

At the present time, the consultant has knowledge of 54 machine types, which account for 90 percent of contract support services activity. For those types not currently included in the knowledge base, the consultant accepts the user's estimate of the time required to perform the service requested. We plan to monitor usage and add rules to cover additional machine types as required.

Concluding remarks

We believe that the consultant, which makes the knowledge of experts widely available, demonstrates that we can increase productivity, provide increased consistency, and improve accuracy in contract management. There are many other places in business, including estimating and proposal preparation, diagnostic tasks, and configuration tasks, that can benefit from the application of knowledge-based expert systems. We hope that this paper stimulates the exploration of such uses to improve operations in these other areas.

Acknowledgments

The authors would like to acknowledge contributions to this paper by the following people: M. L. Burnett, A. W. Elwood, Jr., P. S. Madritch, N. J. Peltier, B. A. Plum, and R. J. Quinn. The project would not have been possible without the support of the management team: C. P. Biggar, R. F. Casella, G. D. Cole, T. A. Ewing, J. R. Fyffe, L. A. Gilmore, J. L. Lucas, and W. C. Richie.

Cited reference

1. B. G. Buchanan and E. H. Shortliffe, *Rule-Based Expert Systems*, Addison-Wesley Publishing Co., Menlo Park, CA (1984).

General references

IBM Expert System Consultation Environment and Expert System Development Environment General Information Manual, GH20-9597-1, IBM Corporation; available through IBM branch offices.

IBM Expert System Consultation Environment/VM, Version 1 Release 1, Installation Guide, SH20-9605, IBM Corporation; available through IBM branch offices.

IBM Expert System Consultation Environment/VM, Version 1 Release 1, User's Guide, SH20-9606, IBM Corporation; available through IBM branch offices.

IBM Expert System Development Environment/VM, Version 1 Release 1, Installation Guide, SH20-9607, IBM Corporation; available through IBM branch offices.

IBM Expert System Development Environment/VM, Version 1 Release 1, User's Guide, SH20-9608, IBM Corporation; available through IBM branch offices.

IBM Expert System Development Environment/VM, Version 1 Release 1, Reference Guide, SH20-9609, IBM Corporation; available through IBM branch offices.

W. Reitman (editor), *Artificial Intelligence Applications for Business*, Ablex Publishing Co., Norwood, NJ (1984).

P. H. Winston, *Artificial Intelligence*, 2nd Edition, Addison-Wesley Publishing Co., Menlo Park, CA (1984).

James A. Voelker *Information Systems Group, Service Research, Knowledge Based Systems, 2099 Gateway Place, San Jose, California 95110.* Mr. Voelker joined IBM in 1960 as a customer engineer in New York City. In 1968, he became a member of the Field Engineering Education staff in Poughkeepsie, New York, serving as an instructor and course developer. He is a graduate of the IBM Systems Research Institute, where he first became interested in artificial intelligence in general and expert systems in particular. He began his present assignment in May of 1985. Mr. Voelker received a B.S. in business from Marist College, Poughkeepsie, New York, in 1977.

Gordon B. Ratica *Information Systems Group, Service Research, Knowledge Based Systems, 2099 Gateway Place, San Jose, California 95110.* Mr. Ratica joined IBM in 1966 as a customer engineer in Oklahoma City, Oklahoma. In 1974, he moved to the General Products Division, Boulder, Colorado, as an applications programmer. In 1979, he returned to the Field Engineering Division in Tulsa, Oklahoma. Since 1983, Mr. Ratica has been involved with the design and implementation of knowledge-based systems.

Reprint Order No. G321-5270.